Report

Team WDNMD

2020/12/13

library(corrplot)

library(car)

library(knitr)  
library(plot.matrix)  
library(class)

library(MASS)

library(leaps)

library(caret)

library(nnet)

library(dplyr)

library(randomForest)

library(gbm)

library(glmnet)

library(lubridate)

## Preprocessing of the Data

testing <- read.csv("test.csv")  
training <- read.csv("training.csv")  
training$PublishedDate <- mdy\_hm(training$PublishedDate)  
training$month <- month(training$PublishedDate)  
training$day<- day(training$PublishedDate)  
training$hour<- hour(training$PublishedDate)  
training$minute<- minute(training$PublishedDate)  
  
testing$PublishedDate <- mdy\_hm(testing$PublishedDate)  
testing$month <- month(testing$PublishedDate)  
testing$day<- day(testing$PublishedDate)  
testing$hour<- hour(testing$PublishedDate)  
testing$minute<- minute(testing$PublishedDate)  
  
  
set.seed(123456)  
index <- sample(seq\_len(nrow(training)), size = 0.8 \* nrow(training))  
train <- training[index,-c(1,2)]  
train <- na.omit(train)  
test <- training[-index,-c(1,2)]

## Statisctic Model Selection

### GLM

model\_glm <- glm(train$growth\_2\_6 ~ ., data = train)  
yhat.glm <- predict(model\_glm, newdata = test)

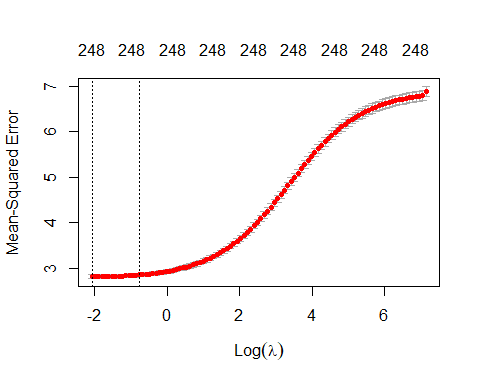
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :  
## prediction from a rank-deficient fit may be misleading

glm.err <- mean((yhat.glm - test$growth\_2\_6)^2)

GLM model MSE: 2.5475668

### Ridge

library(glmnet)  
xtrain <- model.matrix(growth\_2\_6~., data = train)  
ytrain <- train$growth\_2\_6  
xtest <- model.matrix(growth\_2\_6~., data = test)  
ytest <- test$growth\_2\_6  
  
ridge.fit <- cv.glmnet(xtrain,ytrain,alpha = 0)  
plot(ridge.fit)

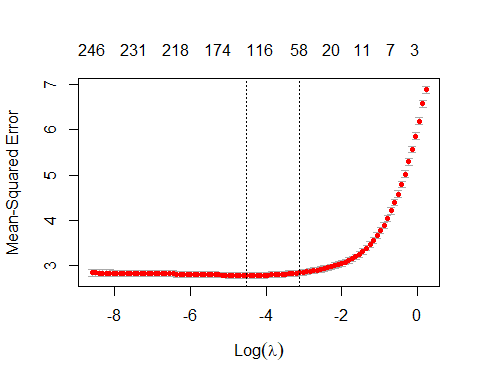


ridge.lambda <- ridge.fit$lambda.min  
  
ridge.pred <- predict(ridge.fit, s = ridge.lambda, newx = xtest)  
ridge.err <- mean((ridge.pred - ytest)^2)

Ridge test MSE: 2.5400127.

### Lasso

lasso.fit <- cv.glmnet(xtrain,ytrain,alpha = 1)  
plot(lasso.fit)



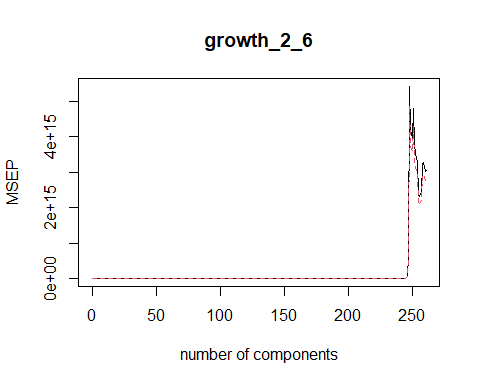
lasso.lambda <- lasso.fit$lambda.min  
#lasso.lambda  
  
lasso.pred <- predict(lasso.fit, s = lasso.lambda, newx = xtest)  
lasso.err <- mean((lasso.pred - ytest)^2)

Lasso test MSE: 2.5599742

### PCR

library(pls)

pcr.fit <- pcr(growth\_2\_6~.,data = train, scale= FALSE, validation = "CV")  
validationplot(pcr.fit, val.type = "MSEP")

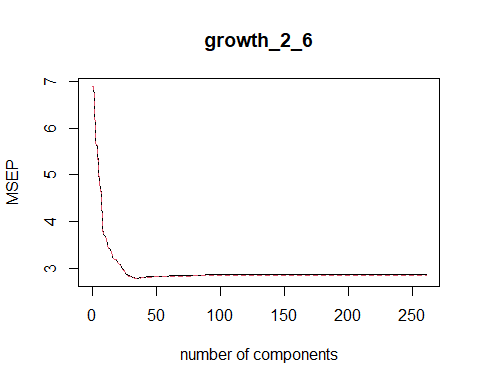


#summary(pcr.fit)  
pcr.pred <- predict(pcr.fit, test, ncomp = 109)  
pcr.err = mean((pcr.pred - test$growth\_2\_6)^2)

PCR test error rate : 2.5657244.

### PLS

pls.fit <- plsr(growth\_2\_6~.,data = train, scale= FALSE, validation = "CV")  
validationplot(pls.fit, val.type = "MSEP")



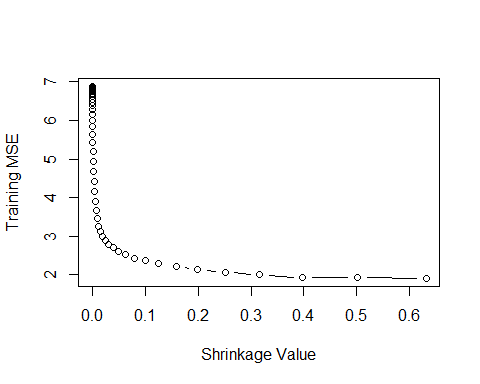
#summary(pls.fit)  
pls.pred <- predict(pls.fit, test, ncomp = 34)  
pls.err = mean((pls.pred - test$growth\_2\_6)^2)

PLS test error rate : 2.5387545.

### Boosting

library(gbm)  
set.seed(123)  
power <- seq(-10, -0.2, by = 0.1)  
lambda <- 10^power  
trainMSE <- rep(NA, length(lambda))  
for (i in 1:length(lambda)){  
 boost <- gbm(growth\_2\_6~., data = train, distribution = "gaussian", n.trees = 500,verbose = FALSE, shrinkage = lambda[i])  
 pred.train <- predict(boost, train, n.trees = 1000)  
 trainMSE[i] <- mean((pred.train - train$growth\_2\_6)^2)  
}

plot(lambda, trainMSE, type = "b", xlab = "Shrinkage Value", ylab = "Training MSE")

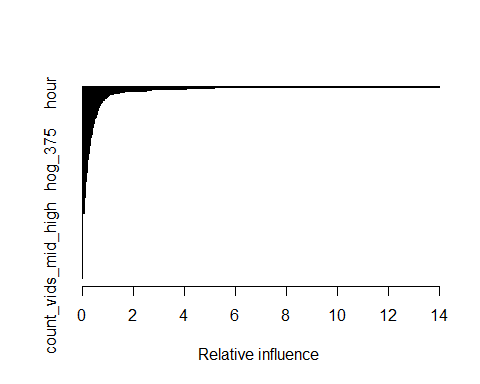


#min(trainMSE)  
#lambda[which.min(trainMSE)]  
  
model\_gbm <- gbm(growth\_2\_6~., data = train, distribution = "gaussian", n.trees = 500, shrinkage = lambda[which.min(trainMSE)])

yhat.gbm <- predict(model\_gbm, newdata = test)

## Using 500 trees...

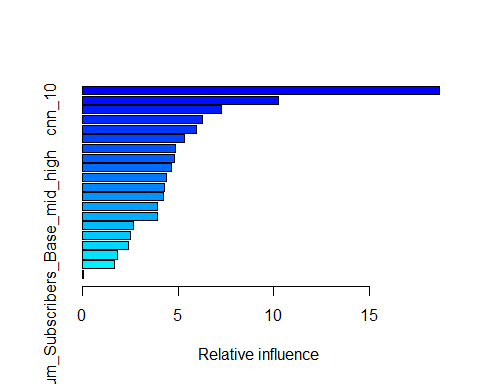
gbm.err <- mean((yhat.gbm - test$growth\_2\_6)^2)   
  
a <- summary(model\_gbm)



gbm\_x <- head(a,20)[,1]  
model\_gbm1 <- gbm(growth\_2\_6~., data = train[,c(gbm\_x, "growth\_2\_6")], distribution = "gaussian", n.trees = 500, shrinkage = lambda[which.min(trainMSE)])  
  
yhat.gbm1 <- predict(model\_gbm1, newdata = test)

## Using 500 trees...

gbm1.err <- mean((yhat.gbm1 - test$growth\_2\_6)^2)   
  
summary(model\_gbm1)



## var rel.inf  
## cnn\_17 cnn\_17 18.64845190  
## Num\_Views\_Base\_mid\_high Num\_Views\_Base\_mid\_high 10.22589007  
## cnn\_10 cnn\_10 7.27887473  
## avg\_growth\_low avg\_growth\_low 6.26273596  
## cnn\_89 cnn\_89 5.94869388  
## pct\_nonzero\_pixels pct\_nonzero\_pixels 5.33038857  
## hog\_643 hog\_643 4.87490171  
## avg\_growth\_low\_mid avg\_growth\_low\_mid 4.82927820  
## num\_words num\_words 4.63475139  
## cnn\_68 cnn\_68 4.39411512  
## views\_2\_hours views\_2\_hours 4.30614076  
## cnn\_25 cnn\_25 4.26066982  
## Duration Duration 3.93968681  
## cnn\_12 cnn\_12 3.91091219  
## hog\_492 hog\_492 2.65284802  
## hour hour 2.50540482  
## cnn\_86 cnn\_86 2.38794966  
## Num\_Subscribers\_Base\_low\_mid Num\_Subscribers\_Base\_low\_mid 1.84216823  
## avg\_growth\_mid\_high avg\_growth\_mid\_high 1.70365198  
## Num\_Subscribers\_Base\_mid\_high Num\_Subscribers\_Base\_mid\_high 0.06248617

Boosted model MSE: 3.078268.

### Random Forest

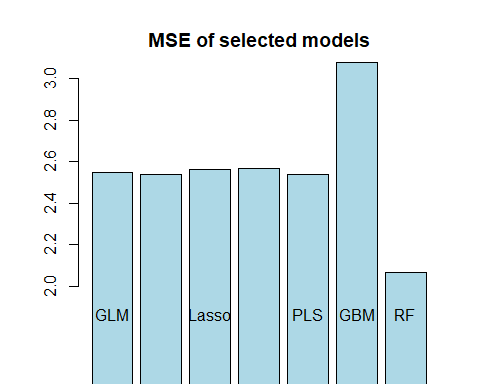
library(randomForest)  
model\_rf <- randomForest(growth\_2\_6~., data = train, mtry = 262/3, ntree= 2000, importance = TRUE) # 2.10

yhatrf <- predict(model\_rf, newdata = test)  
rf.err <- mean((yhatrf - test$growth\_2\_6)^2)

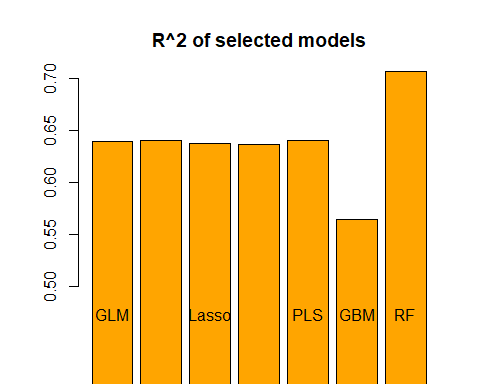
Random forest model MSE: 2.067357.

### Summary

result <- c(glm.err, ridge.err,lasso.err,pcr.err,pls.err,gbm.err, rf.err)  
barplot(result,   
 names.arg = c("GLM", "Ridge","Lasso", "PCR", "PLS", "GBM", "RF"),  
 ylim = c(2,3),   
 col = "lightblue",  
 main = "MSE of selected models",  
 axes = TRUE)



sst <- mean((mean(test$growth\_2\_6) - test$growth\_2\_6)^2)  
  
r2 <- c()  
for ( i in 1:length(result)){  
 r2 <- c(r2, 1 - result[i]/sst)  
}  
barplot(r2,   
 names.arg = c("GLM", "Ridge","Lasso", "PCR", "PLS", "GBM", "RF"),   
 col = "ORANGE",  
 ylim = c(0.5,0.7),   
 main = "R^2 of selected models")



## Predictor Selection

### High correlation

cor <- abs(cor(train$growth\_2\_6,train[,-258]))

## Warning in cor(train$growth\_2\_6, train[, -258]): 标准差为零

pick <- which(cor > 0.2)  
length(pick)

## [1] 19

high\_cor <- colnames(train)[pick]  
  
correlationMatrix <- cor(train[,pick])  
  
  
x <- c(gbm\_x, high\_cor)  
length(x)

## [1] 39

for (i in 1:length(x)){  
 for (j in 1:(i-1)) {  
 if (x[i] == x[j]){  
 x[i] = 0  
 break  
 }  
 }  
}  
x <- x[-which(x == 0)]  
x

## [1] "cnn\_17" "avg\_growth\_low"   
## [3] "avg\_growth\_low\_mid" "cnn\_10"   
## [5] "cnn\_89" "num\_words"   
## [7] "Num\_Subscribers\_Base\_mid\_high" "views\_2\_hours"   
## [9] "hour" "cnn\_12"   
## [11] "Duration" "Num\_Subscribers\_Base\_low\_mid"   
## [13] "cnn\_68" "cnn\_86"   
## [15] "avg\_growth\_mid\_high" "hog\_643"   
## [17] "hog\_492" "cnn\_25"   
## [19] "pct\_nonzero\_pixels" "doc2vec\_17"   
## [21] "num\_chars" "num\_uppercase\_chars"   
## [23] "Num\_Subscribers\_Base\_low" "Num\_Views\_Base\_low"   
## [25] "Num\_Views\_Base\_low\_mid" "Num\_Views\_Base\_mid\_high"   
## [27] "count\_vids\_mid\_high"

model\_1.1 <- randomForest(growth\_2\_6~., data = train[,c(x,"growth\_2\_6")], mtry = 262/3, ntree = 500)

## Warning in randomForest.default(m, y, ...): invalid mtry: reset to within valid  
## range

summary(model\_1.1)

## Length Class Mode   
## call 5 -none- call   
## type 1 -none- character  
## predicted 5793 -none- numeric   
## mse 500 -none- numeric   
## rsq 500 -none- numeric   
## oob.times 5793 -none- numeric   
## importance 27 -none- numeric   
## importanceSD 0 -none- NULL   
## localImportance 0 -none- NULL   
## proximity 0 -none- NULL   
## ntree 1 -none- numeric   
## mtry 1 -none- numeric   
## forest 11 -none- list   
## coefs 0 -none- NULL   
## y 5793 -none- numeric   
## test 0 -none- NULL   
## inbag 0 -none- NULL   
## terms 3 terms call

yhat.1.1 <- predict(model\_1.1, newdata = test)  
mse1.1 <- mean((yhat.1.1 - test$growth\_2\_6)^2)  
mse1.1

## [1] 2.089632

MSE: 2.0896315

### Importance

summary(importance(model\_rf))

## %IncMSE IncNodePurity   
## Min. : -2.643 Min. : 0.00   
## 1st Qu.: 2.318 1st Qu.: 40.38   
## Median : 4.566 Median : 55.47   
## Mean : 9.625 Mean : 151.59   
## 3rd Qu.: 7.066 3rd Qu.: 80.71   
## Max. :129.356 Max. :7007.10

rf\_imp <- which(importance(model\_rf)[,1]>mean(importance(model\_rf)[,1])& importance(model\_rf)[,2]>mean(importance(model\_rf)[,2]))  
rf\_imp <- rownames(importance(model\_rf))[rf\_imp]  
  
set.seed(123)  
model\_1.2 <- randomForest(growth\_2\_6~., data = train[,c(rf\_imp,"growth\_2\_6")], mtry = 262/3, ntree = 500) # 1.988

yhat.1.2 <- predict(model\_1.2, newdata = test)  
mse1.2 <- mean((yhat.1.2 - test$growth\_2\_6)^2)  
mse1.2

## [1] 1.984859